CLAIMS

What is claimed is:

1. A compiler device for optimizing a program which manipulates a character string, the compiler device comprising:

an append instruction detection unit for detecting an append instruction to append a character string to a string variable for storing a character string, in the program;

a store code generation unit for generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit, a store code for storing data of an appendant character string to be appended to the string variable by the append instruction into a buffer, the plurality of append instructions appending the character strings to the same string variable; and

an append code generation unit for generating an append code for appending a plurality of the appendant character strings to the string variable, at a position to be executed before an instruction to refer to the string variable in the program.

2. The compiler device according to claim 1, further comprising:

a reference instruction detection unit for detecting a reference instruction which first refers to the string variable after the character strings have been appended to the string variable by the plurality of append instructions, wherein the append code generation unit generates the append code at a position to be executed after the store codes and before the reference instruction.

3. The compiler device according to claim 1, wherein the append instruction detection unit detects, as the append instruction, a combination of:

an instruction to convert an immutable string variable in which a process of appending a character string is not allowed, into a mutable string variable in which a process of appending a character string is allowed;

an instruction to append the appendant character string to the mutable string variable; and

an instruction to convert the mutable string variable into the immutable string variable.

4. A compiler device for optimizing a program which manipulates a character string, the compiler device comprising:

an append instruction detection unit for detecting an append instruction to append a character string to a string variable for storing a character string, in the program;

a store code generation unit for generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit, a store code for storing an address in memory where an appendant character string to be appended to the string variable by the append instruction is stored, into a buffer, the plurality of append instructions appending character strings to the same string variable; and

an append code generation unit for generating an append code for appending a plurality of the appendant character strings stored in a plurality of the addresses, to the string variable, at a position to be executed before an instruction to refer to the string variable in the program.

5. A compiler device for optimizing a program which manipulates a character string, the compiler device comprising:

a mutable-to-immutable conversion instruction detection unit for detecting a mutable-to-immutable conversion instruction to convert a mutable string variable in which a process of appending a character string is allowed, into an immutable string variable in which a process of appending a character string is not allowed;

an immutable-to-mutable conversion instruction detection unit for detecting an immutable-to-mutable conversion instruction to convert the immutable string variable into the mutable string variable; and

an instruction elimination unit for eliminating the immutable-to-mutable conversion instruction and for causing the mutable string variable as a source variable of the mutable-to-immutable conversion instruction, to be used as the mutable string variable obtained from conversion by the immutable-to-mutable conversion instruction after the immutable-to-mutable conversion instruction, if an instruction to be executed between the mutable-to-immutable conversion instruction and the immutable-to-mutable conversion instruction does not modify a character string stored in the mutable string variable as the source variable of the mutable-to-immutable conversion instruction, and if an instruction to be executed between the

immutable-to-mutable conversion instruction and use of the mutable string variable obtained from the conversion by the immutable-to-mutable conversion instruction does not modify any of the mutable string variable as the source variable of the mutable-to-immutable conversion instruction and the mutable string variable obtained from the conversion by the immutable-to-mutable conversion instruction.

6. The compiler device according to claim 5, wherein the instruction elimination unit further eliminates the mutable-to-immutable conversion instruction if a character string stored in the immutable string variable is not referred to.

7. The compiler device according to claim 6, wherein the instruction elimination unit moves the mutable-to-immutable conversion instruction to each branch destination of a branch instruction to be executed after the mutable-to-immutable conversion instruction, and executes partial dead assignment elimination for eliminating the mutable-to-immutable conversion instruction if a character string stored in the immutable string variable as a destination variable of the mutable-to-immutable conversion

instruction is not referred to on each branch destination of the branch instruction.

8. The compiler device according to claim 5, wherein the immutable-to-mutable conversion instruction detection unit detects, as the immutable-to-mutable conversion instruction, a combination of:

an instruction to reserve a memory area to be used as a mutable string variable; and

an instruction to append a character string stored in the immutable string variable to the mutable string variable.

9. The compiler device according to claim 5, further comprising:

a partial redundancy elimination unit for executing a partial redundancy elimination process of moving the immutable-to-mutable conversion instruction detected by the immutable-to-mutable conversion instruction detection unit to each control flow edge which merges into a single control flow before the immutable-to-mutable conversion instruction,

wherein the instruction elimination unit eliminates the

immutable-to-mutable conversion instruction, if an instruction to be executed between the mutable-to-immutable conversion instruction and the immutable-to-mutable conversion instruction does not modify a character string stored in the mutable string variable as the source variable of the mutable-to-immutable conversion instruction and if an instruction to be executed between the immutable-to-mutable conversion instruction and the use of the mutable string variable obtained from the conversion by the immutable-to-mutable conversion instruction does not modify any of the mutable string variable as the source variable of the mutable-to-immutable conversion instruction and the mutable string variable obtained from the conversion by the immutable-to-mutable conversion instruction, in a program obtained after the partial redundancy elimination process has been executed.

10. The compiler device according to claim 9,

wherein the instruction elimination unit moves the mutable-to-immutable conversion instruction to each branch destination of a branch instruction to be executed after the mutable-to-immutable conversion instruction, and executes partial dead assignment elimination for eliminating the mutable-to-immutable conversion instruction if a character

string stored in the immutable string variable as a destination variable of the mutable-to-immutable conversion instruction is not referred to on each branch destination of the branch instruction.

11. A compiler program for optimizing a program which manipulates a character string, by using a computer, the compiler program causing the computer to function as:

an append instruction detection unit for detecting an append instruction to append a character string to a string variable for storing a character string, in the program;

a store code generation unit for generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit, a store code for storing data of an appendant character string to be appended to the string variable by the append instruction into a buffer, the plurality of append instructions appending the character strings to the same string variable; and

an append code generation unit for generating an append code for appending a plurality of the appendant character strings to the string variable, at a position to be executed before an instruction to refer to the string variable in the

program.

12. A compiler program for optimizing a program which manipulates a character string, by using a computer, the compiler program causing the computer to function as:

an append instruction detection unit for detecting an append instruction to append a character string to a string variable for storing a character string, in the program;

a store code generation unit for generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit, a store code for storing an address in memory where an appendant character string to be appended to the string variable by the append instruction is stored, into a buffer, the plurality of append instructions appending the character strings to the same string variable; and

an append code generation unit for generating an append code for appending a plurality of the appendant character strings stored in a plurality of the addresses, to the string variable, at a position to be executed before an instruction to refer to the string variable in the program.

13. A compiler program for optimizing a program which

manipulates a character string, by using a computer, the compiler program causing the computer to function as:

a mutable-to-immutable conversion instruction detection unit for detecting a mutable-to-immutable conversion instruction to convert a mutable string variable in which a process of appending a character string is allowed, into an immutable string variable in which a process of appending a character string is not allowed;

an immutable-to-mutable conversion instruction detection unit for detecting an immutable-to-mutable conversion instruction to convert the immutable string variable into the mutable string variable; and

an instruction elimination unit for eliminating the immutable-to-mutable conversion instruction and for causing the mutable string variable as a source variable of the mutable-to-immutable conversion instruction, to be used as the mutable string variable obtained from conversion by the immutable-to-mutable conversion instruction after the immutable-to-mutable conversion instruction, if an instruction to be executed between the mutable-to-immutable conversion instruction and the immutable-to-mutable conversion instruction does not modify a character string stored in the mutable string variable as the source variable

of the mutable-to-immutable conversion instruction and if an instruction to be executed between the immutable-to-mutable conversion instruction and use of the mutable string variable obtained from the conversion by the immutable-to-mutable conversion instruction does not modify any of the mutable string variable as the source variable of the mutable-to-immutable conversion instruction and the mutable string variable obtained from the conversion by the immutable-to-mutable conversion instruction.

14. A recording medium having any a compiler program according to claim 11 recorded thereon.

15. A computer program product comprising a computer usable medium having computer readable program code means embodied therein for causing a compiler device, the computer readable program code means in said computer program product comprising computer readable program code means for causing a computer to effect the functions of claim 1.

16. A computer program product comprising a computer usable medium having computer readable program code means embodied therein for causing a compiler device, the computer readable

program code means in said computer program product comprising computer readable program code means for causing a computer to effect the functions of claim 4.

17. A computer program product comprising a computer usable medium having computer readable program code means embodied therein for causing a compiler device, the computer readable program code means in said computer program product comprising computer readable program code means for causing a computer to effect the functions of claim 5.

18. A computer program product comprising a computer usable medium having computer readable program code means embodied therein for causing a compiler device, the computer readable program code means in said computer program product comprising computer readable program code means for causing a computer to effect the functions of claim 11.

19. A method for optimizing a program which manipulates a character string, the method comprising:

detecting an append instruction to append a character string to a string variable for storing a character string, in the program;

generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit, a store code for storing data of an appendant character string to be appended to the string variable by the append instruction into a buffer, the plurality of append instructions appending the character strings to the same string variable; and

generating an append code for appending a plurality of the appendant character strings to the string variable, at a position to be executed before an instruction to refer to the string variable in the program.

20. A method according to claim 19, further comprising:

detecting a reference instruction which first refers to the string variable after the character strings have been appended to the string variable by the plurality of append instructions, wherein the append code generation unit generates the append code at a position to be executed after the store codes and before the reference instruction.

21. An article of manufacture comprising a computer usable medium having computer readable program code means embodied therein for causing optimization of a program which

manipulates a character string, the computer readable program code means in said article of manufacture comprising computer readable program code means for causing a computer to effect the steps of claim 19.

22. A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for optimizing a program which manipulates a character string, said method steps comprising the steps of claim 19.